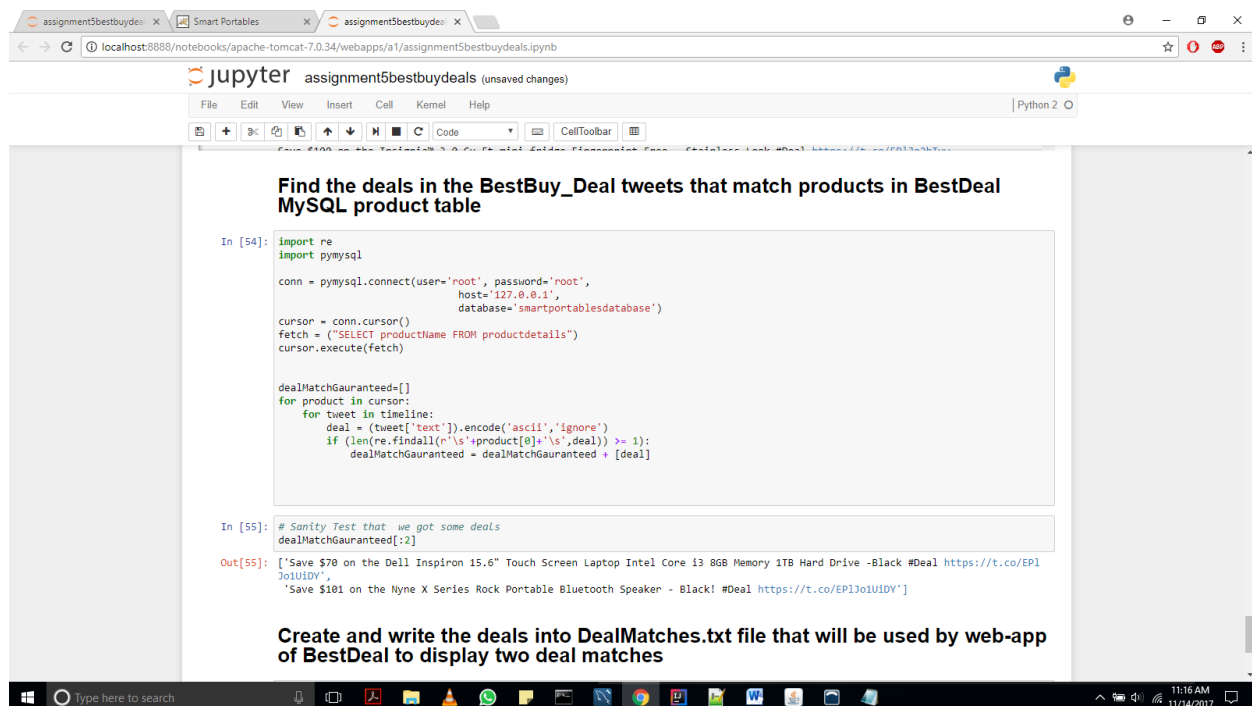


CSP-584 : Enterprise Web Applications Assignment-5

ScreenShot#1 The python script will compare the product names retrieved from MySQL server to the products names retrieved from screen_name BestBuy_Deals on Twitter server and it will write the tweets that have the products names that match the product names retrieved from MySQL server in the product table into DealMatches.txt file.

Python Script for fetching tweets and storing it in txt file after comparing from MySQL productdetails table



The screenshot shows a Jupyter Notebook titled "assignment5bestbuydeals (unsaved changes)" running on a local host. The notebook contains a Python script that connects to a MySQL database, fetches product names from the "productdetails" table, and compares them with tweets from the "BestBuy_Deals" Twitter account. The script identifies two matching deals and prints them out.

```
Find the deals in the BestBuy_Deal tweets that match products in BestDeal MySQL product table

In [54]: import re
import pymysql

conn = pymysql.connect(user='root', password='root',
                        host='127.0.0.1',
                        database='smartportablesdatabase')

cursor = conn.cursor()
fetch = ("SELECT productName FROM productdetails")
cursor.execute(fetch)

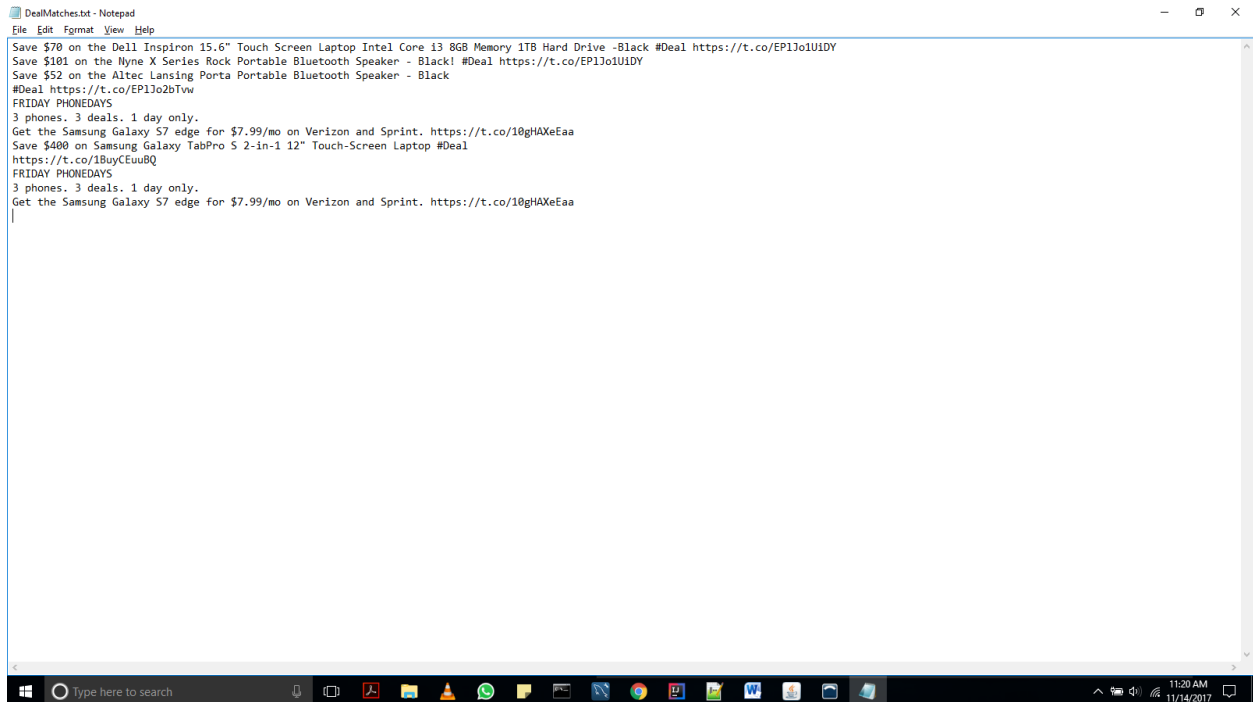
dealMatchGauranteed=[]
for product in cursor:
    for tweet in timeline:
        deal = (tweet['text']).encode('ascii','ignore')
        if (len(re.findall(r'\s'+product[0]+'\\s',deal)) >= 1):
            dealMatchGauranteed = dealMatchGauranteed + [deal]

In [55]: # Sanity Test that we got some deals
dealMatchGauranteed[:2]

Out[55]: ['Save $70 on the Dell Inspiron 15.6" Touch Screen Laptop Intel Core i3 8GB Memory 1TB Hard Drive -Black #Deal https://t.co/EP1JoIUdY',
'Save $101 on the Nyne X Series Rock Portable Bluetooth Speaker - Black! #Deal https://t.co/EP1JoIUdY']

Create and write the deals into DealMatches.txt file that will be used by web-app of BestDeal to display two deal matches
```

DealMatches.txt file

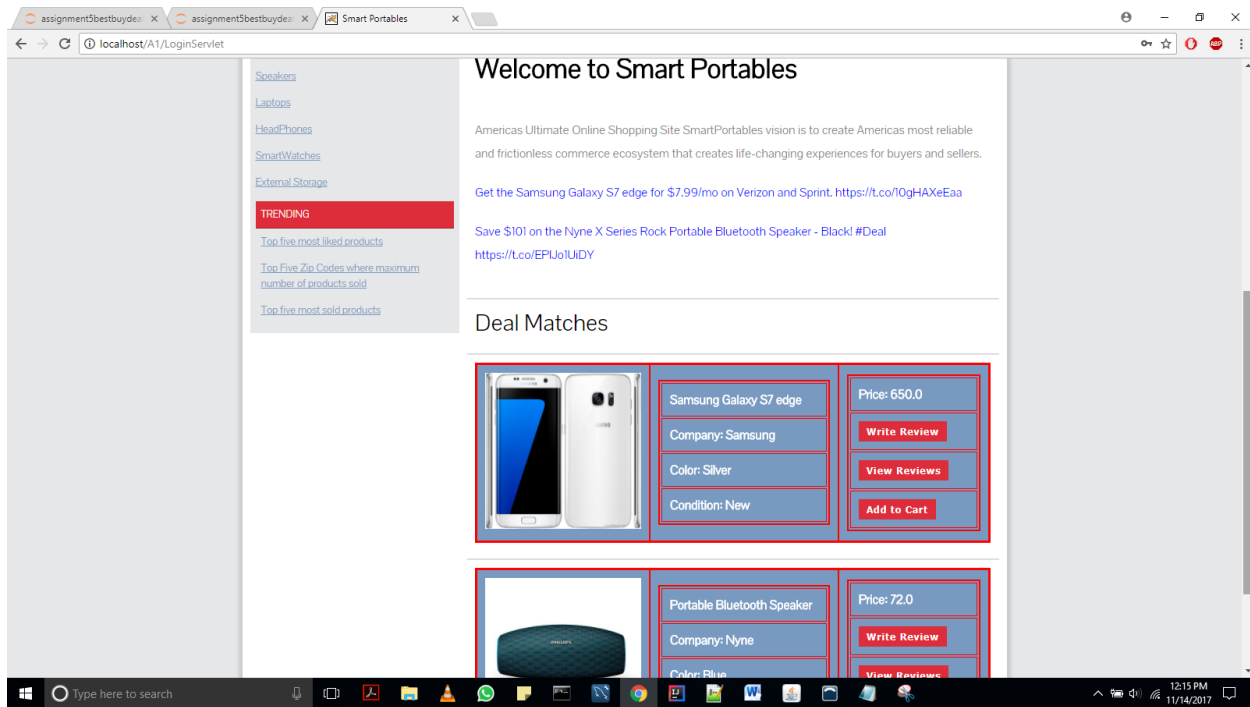


```
DealMatches.txt - Notepad
File Edit Format View Help

Save $70 on the Dell Inspiron 15.6" Touch Screen Laptop Intel Core i3 8GB Memory 1TB Hard Drive -Black #Deal https://t.co/EP1Jo1UIIDY
Save $101 on the Myne X Series Rock Portable Bluetooth Speaker - Black! #Deal https://t.co/EP1Jo1UIIDY
Save $52 on the Altec Lansing Porta Portable Bluetooth Speaker - Black
#Deal https://t.co/EP1Jo2bTvw
FRIDAY PHONEDAYS
3 phones. 3 deals. 1 day only.
Get the Samsung Galaxy S7 edge for $7.99/mo on Verizon and Sprint. https://t.co/10gHAXeEaa
Save $400 on Samsung Galaxy TabPro S 2-in-1 12" Touch-Screen Laptop #Deal
https://t.co/18BuyCEuuBQ
FRIDAY PHONEDAYS
3 phones. 3 deals. 1 day only.
Get the Samsung Galaxy S7 edge for $7.99/mo on Verizon and Sprint. https://t.co/10gHAXeEaa
|
```

ScreenShot#2 When SmartPortables app is started by Tomcat server, it will read ANY 2 Tweets/lines from DealMatches.txt file and display them on SmartPortables app homepage along with links to the individual products that SmartPortables app can match of the offered/displayed deals by BestBuy on the homepage of SmartPortables app.

Below is the screenshot of homepage with two tweets and two products matched from MySQL table Products



Screenshot#3 All new code added for this feature shall be placed in a class called DealMacthes.java

```

1  DealMatches.java
2  import java.util.ArrayList;
3  import java.util.HashMap;
4  import java.io.*;
5  import java.util.*;
6
7  public class DealMatches implements Serializable {
8
9      HashMap<String, Product> productFound;
10     HashMap<String, Product> productHashMap;
11     ArrayList<Object> products;
12     ArrayList<String> twitterTweets = new ArrayList<String>();
13
14     public ArrayList<String> getTwitterTweets()
15     {
16         return twitterTweets;
17     }
18
19     public HashMap<String, Product> getSelectedProductsFromTweets()
20     {
21
22         productFound = new HashMap<String, Product>();
23         products = MySqlDataStoreUtilities.fetchAllProducts();
24         productHashMap = (HashMap<String, Product>)products.get(7);
25         twitterTweets = new ArrayList<String>();
26         String s=null;
27
28         try
29         {
30             for(Map.Entry<String, Product> entry: productHashMap.entrySet())
31             {
32                 if(productFound.size() < 66 && !productFound.containsKey(entry.getKey()))
33                 {
34                     BufferedReader bufferedReader = new BufferedReader(new FileReader(new File("C:\\apache-tomcat-7.0.34\\webapps\\A1\\DealMatches.txt")));
35                     s = bufferedReader.readLine();
36                     if(s==null)
37                     {
38                         else
39                         {
40                             do{
41                                 if(s.contains(entry.getKey()))
42                                 {

```

Screenshot#4 This screenshot is of MySQLDataStoreUtilities.java and is showing the method which fetch all products from productdetails table in MySQL Database

```
DealMatches.java  HomeServlet.java  MySQLDataStoreUtilities.java
281 public static ArrayList<Object> fetchAllProducts()
282 {
283     ArrayList<Object> products = new ArrayList<Object>();
284
285     HashMap<String, Smartphone> smartphones= new HashMap<String, Smartphone>();
286     HashMap<String, Laptop> laptops= new HashMap<String, Laptop>();
287     HashMap<String, Speaker> speakers= new HashMap<String, Speaker>();
288     HashMap<String, Headphone> headphones= new HashMap<String, Headphone>();
289     HashMap<String, Accessory> accessories= new HashMap<String, Accessory>();
290     HashMap<String, Smartwatch> smartwatches= new HashMap<String, Smartwatch>();
291     HashMap<String, WarrantyServlet> warranties= new HashMap<String, WarrantyServlet>();
292
293     HashMap<String, Product> productHashMap= new HashMap<String, Product>();
294
295     Smartphone smartphone;
296     Laptop laptop;
297     Speaker speaker;
298     Headphone headphone;
299     Smartwatch smartwatch;
300     WarrantyServlet warranty;
301     Accessory accessory;
302     Product product;
303
304     try
305     {
306         Connection conn = null;
307         Class.forName("com.mysql.jdbc.Driver").newInstance();
308         conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/smartportablesdatabase?autoReconnect=true&useSSL=false", "root", "root");
309         Statement statement = conn.createStatement();
310         statement.executeQuery("SELECT * FROM productdetails");
311         ResultSet resultSet = statement.getResultSet();
312         while (resultSet.next())
313         {
314             Integer changeId = resultSet.getInt("orderId");
315             String id = changeId.toString();
316             String productType = resultSet.getString("productType");
317             String retailerName = resultSet.getString("retailerName");
318             String imagePath = resultSet.getString("imagePath");
319             String productName = resultSet.getString("productName");
320             String companyName = resultSet.getString("companyName");
```

Screenshot#5 Code for product and tweets displayed HomeServlet.java

```
HomeServlet.java
97 "cp"America's Ultimate Online Shopping Site SmartPortables vision is to create America's most reliable and frictionless commerce ecosystem that creates li
98
99 if(twitterTweets.isEmpty())
100 {
101     pw.println("<p style='color:#0000FF'>"+<No Offers Found>+"</p>");
102 }
103 else
104 {
105     for(String twitter: twitterTweets)
106     {
107         pw.println("<p style='color:#0000FF'>"+twitter+"</p>");
108     }
109 }
110
111 pw.println("</article><article><h2>Deal Matches</h2></article>");
112
113 if(productsFound.isEmpty())
114 {
115     pw.println("<article>");
116     pw.println("<p style='color:#0000FF'>"+<No Deals Found>+"</p>");
117     pw.println("</article>");
118 }
119 else
120 {
121     for(Map.Entry<String,Product> mapping : productsFound.entrySet()){
122
123         Product value = mapping.getValue();
124
125         String productType = value.getType();
126
127         pw.println("<article>");
128         pw.println("<table style='width:100%' style='height:100%' border='2' bordercolor='\"#ff0000\"' cellspacing='\"0\"' cellpadding='\"0\"'>");
129         pw.println("<tr><th width='30%'>");
130         pw.println("<td style='width:200px;height:200px;' style='display:block;'>");
131         pw.println(value.getImage());
132         pw.println("</td>");
133         pw.println("</tr>");
134         pw.println("<tr><td colspan='2'><table border='2' bordercolor='\"#ff0000\"'><tr><th width='40%'><b>");
135         pw.println(value.getName());
136         pw.println("</th><tr><td colspan='2'><tr><td colspan='2'><b>Company:</b>");
```